REMARKS

Applicants have received the Office Action dated April 12, 2007, in which the Examiner: 1) rejected claim 10-14 as allegedly unpatentable under 35 USC 112 1st paragraph 2) rejected claims 10 and 13 under as allegedly anticipated by Gee et al. (U.S. Pat. No. 6,317,872, hereinafter Gee); 2) rejected claims 10-12, 14 and 22-23 as allegedly anticipated by Zaidi (U.S. Pat. No. 6,581,154); 3) rejected claims 18 and 21 as allegedly anticipated by Seal et al. (U.S. Pat. No. 6,965,984, hereinafter Seal); 4) rejected claims 1-4, 7-9 and 19-20 as allegedly obvious over Seal and Gee; 5) rejected claims 5-6 as allegedly obvious over Seal, Gee, and Zaidi; 6) rejected claim 15-17 as allegedly obvious over Seal, Gee, and Greenberger et al. (U.S. Pat. No. 6,317,872, hereinafter Greenberger); and 7) objected to drawings.

I. DRAWING OBJECTIONS

The Office action objects to drawings for failing to show every feature of the invention specified in the claims. In particular, the limitation from claim 10 "the determining independent of the form of the instruction" is allegedly not shown. To address the concerns, Applicants direct the Examiner's attention to Figure 4, the decode logic 152, as well as corresponding paragraph [0025] of the specification. In particular, the decode logic 152, receives instructions from instruction storage via instruction fetch logic, independent of the form of the instruction and decodes instructions to determine the type of instruction (either the Bytecode is execute or replace).

II. 35 U.S.C. § 112 1ST PARAGRAPH BASED REJECTIONS.

Claims 10-14 stand rejected allegedly for lack of enablement.

In the Office action claim 10 stands rejected as allegedly lacking enablement. In particular, the Office action states that one of ordinary skill in the art cannot determine "which bytecodes will be replaced without taking into consideration the form of the instruction." To address the concerns, Applicants direct the Examiner to Figure 4, the decode logic 152, as well as corresponding paragraph [0025] of the specification. The decode logic 152, first, receives instructions from the instruction storage via instruction fetch logic. Second, the received instructions are decoded to determine the type/form of instruction; hence, the determination of the type of instruction is done independent of the form of the instruction fetched. The decode logic is enabled to perform the step of determining independent of the form of instruction received from the storage via the fetch logic.

Based on the foregoing, Applicants respectfully submit that 35 USC 112 1st paragraph lack of enablement rejection be withdrawn.

Claims 10-14 also stand rejected for failing to comply with written description requirement. To address the concerns, Applicants direct the Examiner to Figure 4, the decode logic 152, as well as corresponding paragraph [0025] of the specification. The decode logic 152, decodes the instructions fetched from the storage to determine the type of instruction. The determination of the type of instruction is done independent of the type/form of the instruction fetched.

Based on the foregoing, Applicants respectfully submit that 35 USC 112 1st paragraph lack of written description requirement rejection be withdrawn.

III. ART BASED REJECTIONS

A. Claim 1

Claim 1 stands rejected as allegedly obvious over Seal and Gee.

Seal is directed to data processing using multiple instruction sets. (Seal Title). However, on closer inspection of Seal it is found that the Seal processor does not use multiple instruction sets. Seal's Figure 1 is reproduced immediately below for convenience of the discussion.

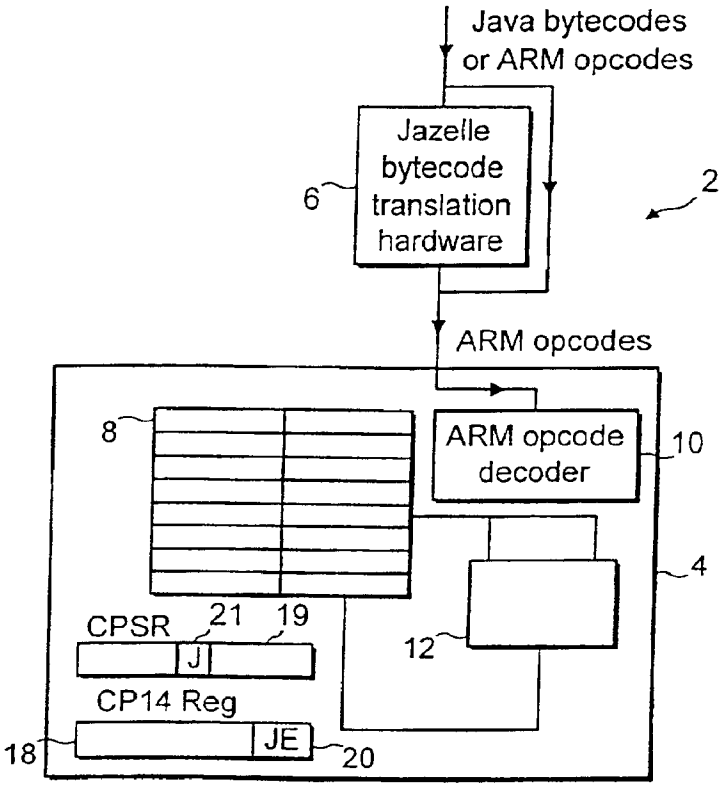

FIG. 1

In particular, in Seal the stream of Java bytecodes are provided to bytecode translation hardware 6. (Seal Col. 6, lines 10-14; Figure 1). For some bytecodes, the bytecode translation hardware 6 generates a series of ARM opcodes which are applied to the ARM opcode decoder 10. (Seal Col. 6, lines 10-29; Figure 1). For other bytecodes, (bytecodes not supported by the hardware) the bytecode translation hardware triggers a software instruction interpreter written in ARM native instructions. (Seal Col. 6, lines 30-39).

Gee is directed to a real time processor optimized for executing Java programs. (Gee Title). In particular, Gee teaches using an Advanced Architecture Microprocessor (AAMP) programmed to simulate direct execution of Java bytecodes by using each bytecode as a pointer to a microinstruction sequence that performs the function of the bytecode. (Gee Col. 8, lines 57-59). Thus, the AAMP knows only one instruction set, those of the microinstructions.

Claim 1, by contrast, specifically recites, "decode logic coupled to said fetch logic, the decode logic decodes instructions from a first instruction set and a second instruction set, the second instruction set different than the first instruction set; and an active program counter selected as either a first program counter or a second program counter; wherein an instruction of the first instruction set is replaced by a micro-sequence comprising one or more instructions of the second instruction set and the active program counter switches between the first and second program counters based on a micro-sequence-active bit." Applicants respectfully submit that Seal and Gee fail to teach or suggest such a processor. In both Seal and Gee, their decoders appear to decode only a single instruction set, with each and every Java bytecode translated to the native instruction set. Thus, Seal and Gee fail to teach or suggest "decode logic coupled to said fetch logic, the decode logic decodes instructions from a first instruction set and a second instruction set, the second instruction set different than the first instruction set."

Based on the foregoing, Applicants respectfully submit that claim 1, and all claims which depend from claim 1 (claims 2-9), should be allowed.

B. Claim 10

Claim 10 stands rejected as allegedly anticipated by Gee, and as allegedly anticipated by Zaidi. Applicants amend claim 10 to make the claim terms more consistent with the specification, and not to define over any cited art.

Gee is directed to a real time processor optimized for executing Java programs. (Gee Title). In particular, in Gee each and every bytecode is used as a pointer to microinstructions. (Gee Col. 8, lines 57-59).

Zaidi is directed to expanding microcode associated with full and partial width macro-instructions. (Zaidi Title). In particular, Zaidi discloses a system where a macro-instruction is used to identify either a single micro-operation or a plurality of micro-operations. (Zaidi Col. 2, lines 55-67). Thus, in Zaidi the macro-instruction is not executed; rather, the macro-instruction identifies either a single micro-operation (uops) or a series of micro-operations (suops) that actually execute. (Zaidi Col. 3, line 59 through Col. 4, line 7). As for the series of micro-operations, the series can be overloaded to be either scalar or packed. (*Id.*) It appears in Zaidi that scalar or packed version to be used is identified as part of the macro-instruction, *e.g.*, ADDS (add scalar) or ADDP (add packed). (Zaidi Col. 3, lines 8-15; Figure 3).

Claim 10, by contrast, specifically recites, "fetching an instruction; and determining whether said instruction is to be executed or replaced by a group of other instructions, the determining independent of the type of the instruction." Applicants respectfully submit that neither Gee nor Zaidi expressly or inherently teach such a system. As for Gee, in Gee each and every bytecode is replaced by a series of microinstructions. Thus, Gee does not "determine[e] whether said instruction is to be executed." Nor is such a determining inherent as suggested by the Office action, as there is no need to make such a determination if each and every bytecode is replaced by a series of microinstructions.

As for Zaidi, in Zaidi the macro-instruction is not executed; rather, the macro-instruction merely identifies either a micro-instruction or a series of micro-instructions to be executed. Thus, Zaidi fails to expressly or inherently teach "fetching an instruction; and determining whether said instruction is to be executed... ." Moreover, even if hypothetically the macro-instruction of Zaidi is the claimed fetched instruction (which Applicants do not admit), the macro-instruction itself appears to identify whether it is scalar or packed, and thus Zaidi fails to teach "the determining independent of the type of the instruction."

Based on the foregoing, Applicants respectfully submit that claim 10, and all claims which depend from claim 10 (claims 11-14) should be allowed.

C. Claim 15

Claim 15 stands rejected as allegedly obvious over Seal, Gee and Greenberger.

Gee is directed to a real time processor optimized for executing Java programs. (Gee Title). In particular, Gee teaches using an Advanced Architecture Microprocessor (AAMP) programmed to simulate direct execution of Java bytecodes by using each bytecode as a pointer to a microinstruction sequence that performs the function of the bytecode. (Gee Col. 8, lines 57-59). Thus, the AAMP knows only one instruction set, those of the microinstructions.

Seal is directed to data processing using multiple instruction sets. (Seal Title). In particular, in Seal the stream of Java bytecodes are provided to bytecode translation hardware 6. (Seal Col. 6, lines 10-14; Figure 1). For some bytecodes, the bytecode translation hardware 6 generates a series of ARM opcodes which are applied to the ARM opcode decoder 10. (Seal Col. 6, lines 10-29; Figure 1). For other bytecodes, the bytecode translation hardware triggers a software instruction interpreter written in ARM native instructions. (Seal Col. 6, lines 30-39).

Claim 15, by contrast, specifically recites, "decode logic coupled to said fetch logic, the decode logic decodes instructions from a first instruction set and a second instruction set, the second instruction set different than the first instruction set....wherein an instruction of the first instruction is replaced by a micro-sequence comprising one or more instructions from the second instruction set." Applicants respectfully submit that Gee and Seal fail to teach or suggest such a system. In both Seal and Gee, their decoders appear to decode only a single instruction set, with each and every Java bytecode translated to the native instruction set. Thus, even if the teachings of Greenberger are precisely as the Office action suggests (which Applicants do not admit), Seal, Gee and Greenberger still fail to teach or suggest "decode logic coupled to said fetch logic, the decode logic decodes instructions from a first instruction set and a second instruction set, the second instruction set different than the first instruction set."

Based on the foregoing, Applicants respectfully submit that claim 15, and all claims which depend from claim 15 (claims 16-17), should be allowed.

D. Claim 18

Claim 18 stands rejected as allegedly anticipated by Seal,

Seal is directed to data processing using multiple instruction sets. (Seal Title). In particular, in Seal the stream of Java bytecodes are provided to bytecode translation hardware 6. (Seal Col. 6, lines 10-14; Figure 1). For some bytecodes, the bytecode translation hardware 6 generates a series of ARM opcodes which are applied to the ARM opcode decoder 10. (Seal Col.

6, lines 10-29; Figure 1). For other bytecodes, the bytecode translation hardware triggers a software instruction interpreter written in ARM native instructions. (Seal Col. 6, lines 30-39).

Claim 18, by contrast, specifically recites, "decode logic that decodes instructions, the decode logic decodes instructions from a first instruction set and a second instruction set, the second instruction set different than the first instruction set." With respect to the anticipation rejection over Seal, Seal appears to use only a single instruction set, the ARM opcodes. Thus, Seal fails to expressly or inherently teach the limitations of claim 18.

Based on the foregoing, Applicants respectfully submit that claim 18, and all claims which depend from claim 18 (claims 19-21), should be allowed.

E. Claim 22

Claim 22 stands rejected as allegedly anticipated Zaidi.

Zaidi is directed to expanding microcode associated with full and partial width macro-instructions. (Zaidi Title). In particular, Zaidi discloses a system where a macro-instruction is used to identify either a single micro-operation or a plurality of micro-operations. (Zaidi Col. 2, lines 55-67). Thus, in Zaidi the macro-instruction is not executed; rather, the macro-instruction identifies either a single micro-operation (uops) or a series of micro-operations (suops) that actually execute. (Zaidi Col. 3, line 59 through Col. 4, line 7). As for the series of micro-operations, the series can be overloaded to be either scalar or packed. (*Id.*) It appears in Zaidi that scalar or packed version to be used is identified as part of the macro-instruction, *e.g.*, ADDS (add scalar) or ADDP (add packed). (Zaidi Col. 3, lines 8-15; Figure 3).

Claim 22, by contrast, specifically recites, "decode logic that decodes instructions; and a means for determining whether an instruction is to be executed or replaced by a micro-sequence of other instructions." Applicants respectfully submit that Zaidi does not expressly or inherently teach such a system. In Zaidi the macro-instruction is not executed; rather, the macro-instruction merely identifies either a micro-instruction or a series of micro-instructions to be executed. Thus, Zaidi fails to expressly or inherently teach "a means for determining whether an instruction is to be executed or replaced by a micro-sequence of other instructions."

Based on the foregoing, Applicants respectfully submit that claim 10, and claim 23 which depend from claim 22, should be allowed.

IV. CONCLUSION

In the course of the foregoing discussions, Applicants may have at times referred to claim limitations in shorthand fashion, or may have focused on a particular claim element. This discussion should not be interpreted to mean that the other limitations can be ignored or dismissed. The claims must be viewed as a whole, and each limitation of the claims must be considered when determining the patentability of the claims. Moreover, it should be understood that there may be other distinctions between the claims and the cited art which have yet to be raised, but which may be raised in the future.

Applicants respectfully request reconsideration and that a timely Notice of Allowance be issued in this case. It is believed that no extensions of time or fees are required, beyond those that may otherwise be provided for in documents accompanying this paper. However, in the event that additional extensions of time are necessary to allow consideration of this paper, such extensions are hereby petitioned under 37 C.F.R. § 1.136(a), and any fees required (including fees for net addition of claims) are hereby authorized to be charged to the Texas Instruments, Inc. Deposit Account No. 20-0668.

Respectfully submitted,

/Utpal D. Shah/

Utpal D. Shah
PTO Reg. No. 60,047
CONLEY ROSE, P.C.
(512) 391-1900 (Phone)
(512) 320-9182 (Fax)
AGENT FOR APPLICANTS